Neural Network

1. 最经典的：全连接神经网络

   所有神经元全连接，激活，传递。一层视作一个函数,网络即一个函数集。

   vector -- vector

   常用矩阵来计算网络传播：

$$x1 = \sigma(Wx + b$$

   网络结构的构建一般是直觉性/经验性的。

   y_hat和y计算损失函数(一般使用交叉熵)，根据loss使用gradient descent, 优化参数。

   Backpropogation：很多的包已经提供了反向传播的自动求导。(toolkit)

2.一些问题：

为什么deep?

①越deep，表现未必越好。一个 hidden layer 的网络能够表示任何的函数。

Gradient Descent

主要数学依据: 链式法则

损失函数定义:

$$\Sigma dist(y_n, y'_n) = L(\Theta)$$

只需要计算对某个参数的偏微分即可。

$$\frac{\partial L(\theta)}{\partial w} = \sum_{n=1}^{N} \frac{\partial C^n(\theta)}{\partial w}$$

根据链式法则传递偏导数，最后得到每个参数对Loss的偏导数，再乘上学习率完成参数优化。

反向传播

规则:一直向前递归求取偏微分结果直到抵达output layer为止。

求法：从后向前传播，其实计算量和从前向后传播的计算量是一样的。(相当于将网络结构反置，建立一个反向的神经网络，每一级的激活函数是原神经元位置函数的导数)